MOBILE ROBOT TRACKING OF PRE-PLANNED PATHS

N. E. Pears

Department of Computer Science, York University, Heslington, York, Y010 5DD, UK (email:nep@cs.york.ac.uk)

Abstract—A method of mobile robot steering control around pre-planned paths is presented. The system can manœuvre accurately at low speeds by deriving control parameters as functions of vehicle velocity. The peak demand on the steering controller is reduced, by distributing steering curvature changes evenly over the extent of a manœuvre.

Key words: Mobile robots; Autonomous vehicles; Steering control; Path tracking; Path following

1. INTRODUCTION

The robotics research community has invested significant effort in a variety of solutions to the problem of mobile robot control. Earlier examples include an optimal proportional-plus-integral controller for a wire-guidance vehicle [4], a method whereby the accelerations of the left and right drive wheels are controlled such that the vehicle's pose will agree with that of the current command after a given time period [3], and an exponential control law to track a sequence of points [6]. More recently, detailed non-linear models have been included in the control analysis [1] and sliding mode control methods have become popular for both trajectory tracking [7] and in the context of artificial potential fields [2]. Other recent research has bypassed the pose estimation phase normally assumed and used sensory information directly for control [5]. Here, we assume sensory pose estimation and that a path has been planned that is consistent with the kinematic constraints of the vehicle. We then go on to develop simple linear state feedback laws and test and augment them for a variety of path complexities. In comparison to many previous systems, the steering control described here is simple, robust and effective due to these intuitive state feedback control laws.

2. THE CONTROL LAW

We assume that there is always an estimate of global vehicle position, (x_g, y_g, θ_g), from some sensor system(s). This estimate may be subject to small step changes due to either bumps on the floor, or new landmarks becoming visible, causing a discontinuity in the estimation. Thus we need to analyse both the transient and steady state behaviour of path tracking performance. In addition to a global position estimate, we require some definition of the planned path within the

global frame. We then determine the position of a local frame situated on the path, whose y axis is perpendicular to the path and incident with the robot reference point (see figure 1). This local frame moves along the path at a speed equal to the tangential velocity of the robot relative to the path. Measurement of the vehicle position in this local frame gives a normal offset error, y, and orientation error, θ, relative to the path. Assuming that the vehicle velocity along the specified path can be controlled adequately, we now present a steering control law, which forces the robot's reference point to track the demand trajectory.

Since vehicle heading, θ, in the local frame affects the way in which tracking error, y, changes, the control algorithm initially generates a demand heading relative to the orientation of the local frame. This always points towards the x axis of the local frame at an angle that is proportional to the tracking error and in the direction that the robot is travelling. The error in robot heading with respect to a demand heading derived in this fashion yields a proportional turning curvature that tends to diminish this heading error. In effect, the robot can turn left and then right by steering around each of the demand heading lines shown in figure 1.

3. KINEMATIC ANALYSIS

Initially we assume ideal dynamics of the steering controller, which allows the robot to instantaneously assume any arbitrary turning curvature. The control law can be formally specified as follows: Generate a demand heading in the local frame, θ_d, proportional to the path offset error, so that:

$$\theta_d = -k_y(y - y_d) \tag{1}$$

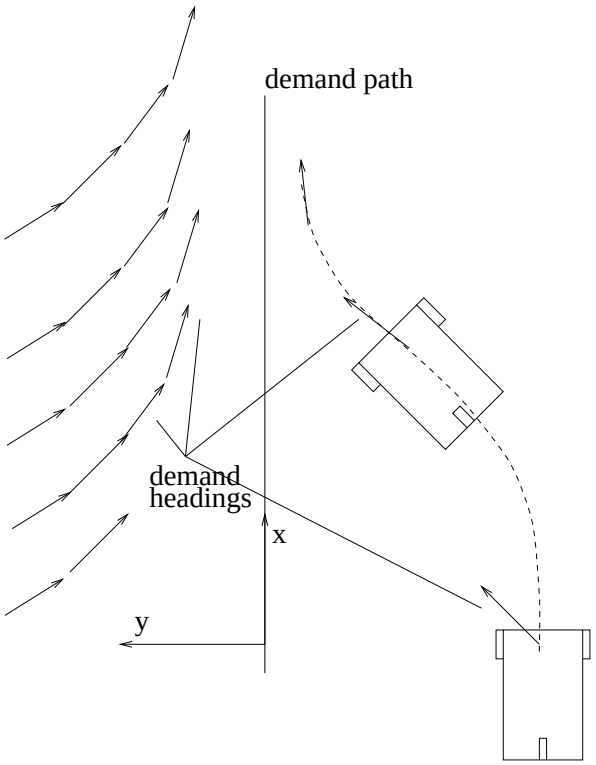

Figure 1: The control law

subject to $|\theta_{d_{max}}| = \frac{\pi}{2}$. (Note that the demand offset from the path, y_d, is zero, but is included so that we can generate a transfer function later.) Then, generate a steering curvature, κ, which is proportional to the heading error:

$$\kappa = -k_\theta(\theta - \theta_d) \tag{2}$$

where k_y and k_θ are constants. Combining controller equations 1 and 2 gives:

$$\kappa = -k_y k_\theta(y - y_d) - k_\theta\theta \tag{3}$$

Let the system state be the vehicle's position in the path dependent local frame: $\mathbf{x^T} = [\mathbf{y}, \theta]$. Kinematic relations for the derivatives of the state variables are:

$$\dot{y} \approx V\theta \tag{4}$$

(small angle approximation), and

$$\dot{\theta} = V\kappa \tag{5}$$

We can now write equations 3, 4 and 5 in the linear form $\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{b}\mathbf{u}$

$$\begin{bmatrix} \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} 0 & V \\ -k_\theta k_y V & -k_\theta V \end{bmatrix} \begin{bmatrix} y \\ \theta \end{bmatrix} + \begin{bmatrix} 0 \\ k_\theta k_y V \end{bmatrix} y_d \tag{6}$$

The characteristic equation of the above system is given by

$$|\mathbf{A} - s\mathbf{I}| = 0 \tag{7}$$

which in this case is:

$$s^2 + k_\theta V s + k_\theta k_y V^2 = 0 \tag{8}$$

where, s, is the variable of the Laplace transform. Note that if the robot is travelling backwards, speed, V, is negative, so both k_θ and k_y must be made negative to keep coefficients positive and the system stable (Routh-Hurwitz criterion). The characteristic equation of the system (equation 8) can be written in the normalised form:

$$\frac{s^2}{\omega^2} + Ts + 1 = 0 \tag{9}$$

with

$$T = \frac{1}{k_y V}, \quad \omega = \sqrt{k_\theta k_y} V \tag{10}$$

The damping factor, ξ, is:

$$\xi = \frac{T\omega}{2} = \frac{1}{2}\sqrt{\frac{k_\theta}{k_y}} \tag{11}$$

Thus to give critical damping (fastest response without overshoots), we require $\xi = 1$ or

$$k_\theta = 4k_y \tag{12}$$

The closed loop transfer function of the system can be derived from the state space matrices as

$$G_{cl}(s) = \frac{Y(s)}{Y_d(s)} = \mathbf{c}[s\mathbf{I} - \mathbf{A}]^{-1}\mathbf{b} \tag{13}$$

where $\mathbf{c} = [1, 0]$ is the matrix that relates the system states to the output, y, and $\mathbf{A}$ and $\mathbf{b}$ are matrices as defined in equation 6. Hence

$$G_{cl}(s) = \frac{k_\theta k_y V^2}{s^2 + k_\theta V s + k_\theta k_y V^2} \tag{14}$$

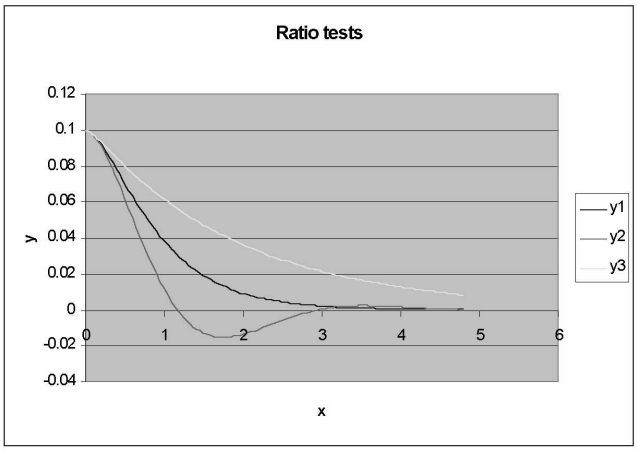

Figure 2: Step responses for various control parameters

The open loop transfer function of the system is related to the closed loop transfer function by

$$G_{ol}(s) = \frac{G_{cl}(s)}{1 - G_{cl}(s)} = \frac{k_y k_\theta V^2}{s(s + k_\theta V)} \tag{15}$$

3.1 Simulation results

Damping factor (equation 11), shows that if $k_\theta > 4k_y$ the robot's motion is overdamped and if $k_\theta < 4k_y$ the motion is underdamped. This is confirmed in figure 2 which shows simulation results of the distance, y, to the demand path as the vehicle attempts to track a straight line. In the experiments, the robot was placed parallel to its demand path, but with a 0.1m offset. Graphs y_1, y_2 and y_3 show the paths taken with control constants (k_θ, k_y) set at (4,1), (2,2) and (8,0.5) respectively.

4. TRACKING CIRCLES

Simulation results for tracking a circle of radius 1m, with control constants set at $k_\theta = 4, k_y = 1$. show that a large steady state offset of around 0.2m quickly develops and the control system, as it stands, is clearly inadequate for curve tracking. In steady state, the system *type* dictates that the vehicle travels parallel to its demand path. (Since it indicates a finite, bounded steady state error.) This implies that the steady state value of local heading, θ_{ss}, is zero. Thus, from equation 3 we have

$$\kappa_{ss} = -k_y k_\theta y_{ss} \tag{16}$$

We also note that the steady state curvature is given as $\kappa_{ss} = \frac{1}{r_{path} - y_{ss}}$ where r_{path}, is the radius of the circular demand path. Substituting for κ_{ss} and rearranging gives the quadratic

$$y_{ss}^2 - r_{path} y_{ss} - \frac{1}{k_\theta k_y} = 0 \tag{17}$$

Equation 17 indicates that the steady state offset in the state y, y_{ss}, is dependent on the path curvature ($\kappa_{path} = \frac{1}{r_{path}}$) and the choice of control constants. Substituting values of $r_{path} = 1.0$, $k_\theta = 4$, and $k_y = 1$ gives a solution to equation 17 as $y_{ss} = -20.0cm$, which is in close agreement with the value obtained from the simulation.

Adding κ_{path} to the right hand side of equation 16 allows y_{ss} to be zero when $\kappa_{ss} = \kappa_{path}$. If κ_{path} is considered as an input, then this curvature is added to that derived from the measured local position (y, θ') and the robot accurately tracks the circular demand path. The simplified overview of this system is shown in figure 3.

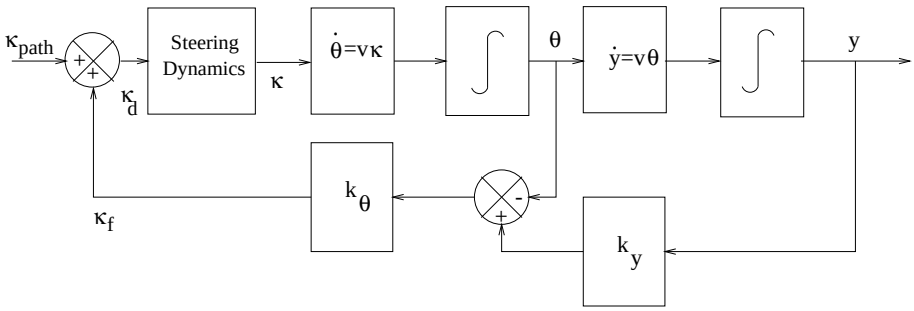

Figure 3: Steering control system

5. VELOCITY DEPENDENCIES

In accordance with the common experience of driving a car, the robot should slow down for sharp corners, allowing safe, smooth and accurate tracking. To implement this we generate a *demand* velocity, V_d, for the speed controller, which is a function of turning curvature such that

$$V_d(\kappa) = \frac{\dot{\theta}_{max}}{\kappa_{lim}}, \quad \kappa_{lim} = max\{\kappa, \frac{\dot{\theta}_{max}}{V_{max}}\} \tag{18}$$

where $\dot{\theta}_{max}$ and V_{max} are constants. Actual vehicle follows this demand velocity as closely as possible in accordance with the speed controller and vehicle dynamics. The control constants, (k_y, k_θ) are computed as a function of this *actual* velocity, so that the vehicle can turn more sharply towards the demand path at lower speeds. This requirement may be formulated by making the system's performance, described by T and ξ in equations 10 and 11 (or equivalently the coefficients in the characteristic equation) independent of velocity. Thus, for velocity independent rise time and unity damping ratio, we require:

$$k_y = \frac{\gamma}{v_{lim}}, \quad k_\theta = \frac{4\gamma}{v_{lim}} \tag{19}$$

$$V_{lim} = max\left\{V, \frac{\gamma}{k_{y_{max}}}\right\} \tag{20}$$

where typically we have used $\gamma = 0.2$ and $k_{y_{max}} = 16$.

This constant rise time feature of the control system means that, when travelling more slowly, any deviations from the planned path are eliminated over a shorter path length. This is beneficial in two instances

- As the robot approaches its destination, it decelerates to a low velocity. This means that raised values of the control constants allow more accurate docking manœvres.

- High curvatures in planned paths typically occur when manœvering close to objects. In such cases, the vehicle velocity is automatically lowered, allowing raised values of the control constants to give accurate tracking, when subject to disturbances such as bumps in the floor.

6. CURVATURE RATE LIMITING

In the preceding sections, we assumed that the steering controller could instantaneously assume any desired curvature. The resulting analysis yielded a 4:1 ratio between control constants for critically damped response. The graph labelled *"curv raw"* on figure 4 shows the curvature required to correct an error in y when the vehicle is parallel to a stright path but displaced 0.1m from it. The graph indicates that a large initial step change in curvature is required followed by a sequence of much smaller changes. In real systems, the steering controller will take a finite time to attain this step in demanded curvature, so we must either:

1. Include the dynamics of the steering process in the analysis and recompute the optimum ratio of control constants.

2. Include in the system a mechanism to spread the change in curvature more evenly over the whole steering process, and assume that, to an acceptable accuracy, the real steering mechanism can track the demanded turning curvature.

To avoid assuming specific steering control dynamics, here we present a solution based on the second approach.

6.1 Filtering demand curvature

In order to spread the changes in curvature more evenly across the whole manœuvre, the actual curvature passed to the steering controller is the value of the previous curvature plus a fraction of the difference between the demand curvature and previous curvature.

$$\kappa_i = \kappa_{i-1} + k_l(\kappa_{d_i} - \kappa_{i-1}) = (1 - k_l)\kappa_{i-1} + k_l\kappa_{d_i} \tag{21}$$

where k_l is a constant such that $\{0 < k_l <= 1\}$. In the z transform domain (where $z = e^{sT_s}$), the transfer function of such a filter is given by

$$G_l(z) = \frac{\kappa_i}{\kappa_{dem_i}} = \frac{k_l z}{z - (1 - k_l)} \tag{22}$$

Equation 21 has the form of a backward difference integration, in which the function integrated is the difference between the demand turning curvature and the curvature actually implemented by the vehicle's wheel(s). The backward difference mapping between the z and s domains can be written as

$$G_l(s) = G_l(z)|_{z=(1-T_s s)^{-1}} \tag{23}$$

and using this mapping on equation 22 gives:

$$G_l(s) = \frac{k_l}{k_l + (1-k_l)T_s s} \tag{24}$$

which can be written as

$$G_l(s) = \frac{k'_l}{s + k'_l}, \quad k'_l = \frac{k_l}{(1-k_l)T_s} \tag{25}$$

The introduction of this filter, modifies the system's open loop transfer function (equation 15) to:

$$G_{ol}(s) = \frac{k_\theta k_y V^2 k'_l}{s(s + k_\theta V)(s + k'_l)} \tag{26}$$

6.2 Simulation results

The same step response test was implemented as in section , except that the curvature filter was included before implementing curvature on the simulated vehicle. Figures 4 and 5 compare the results of turning curvature, κ, and offset from the demand path, y, for the limited ($k_l = 0.1$) and unlimited ($k_l = 1.0$) system when $k_y = 1.0, k_\theta = 4, T_s = 0.1s$ and $V = 0.2s$.

Notice that the step responses in y are very similar in form, as the time constant of the curvature filter is short compared to the time for the vehicle manœuvre. However, the form of κ for the curvature limited system is significantly preferable to the unlimited system. In the limited system, the initial rate of change of curvature required is 10% of the unlimited system, and the changes in curvature are distributed much more evenly over the whole of the manœuvre.

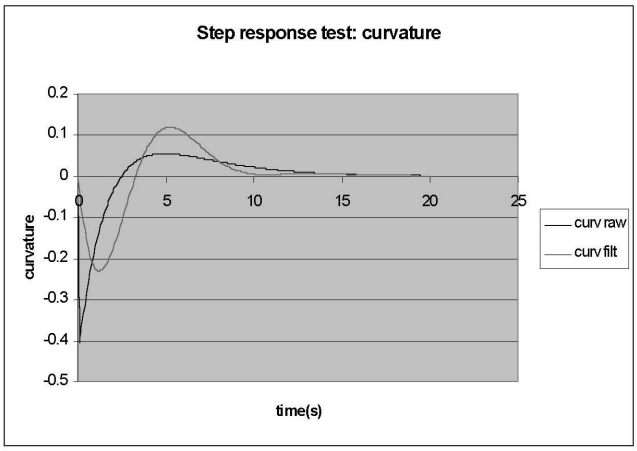

Figure 4: Step response (κ) with curvature rate limiting

A root locus plot employing the modified open loop transfer function in equation 26 indicates that the system is now underdamped. (As expected from the introduction of a lag into the system.) If necessary, this may be restored by either increasing k_{theta} or decreasing k_y. Although the latter option reduces rise time, it is preferable as increasing k_θ. Simulations have indicated that an increase in k_θ can have the undesirable effect of introducing oscillations (multiple sign changes) into κ, even though y does not change sign.

7. TRACKING PARAMETRIC CURVES

Although paths could be constructed with straight line and curve segments, this offers little flexibility. Also, at the junction points between path segments, there would be a discontinuity in curvature which can not be tracked by real steering mechanisms. Parametric curves, such as cubic B-spline are often used to represent planned paths as they offer an attractive trade-off between the flexibility of higher order curves and the control and simplicity of lower order

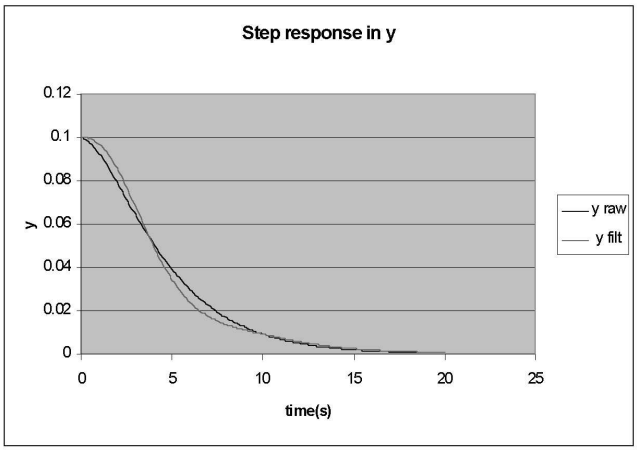

Figure 5: Step response (y) with curvature rate limiting

curves. Also, they are formulated to give continuity up to the second derivative with respect to their parameter, s, [1] which implies continuity in curvature at the knot points (segment junctions).

As the vehicle attempts to track a B-spline, we need to determine the curve parameter value, s, which corresponds to the closest Euclidean distance from the path to the robot. A parametric 2D curve has the form $\mathbf{p}(s) = (x(s), y(s))$. To determine a location on the curve where the robot's position is normal to that curve, we require that

$$\mathbf{p}'.(\mathbf{p_g} - \mathbf{p}) = 0 \tag{27}$$

where $\mathbf{p_g} = (x_g, y_g)$ is the global position of the robot. (Primed values represent differentiation wrt s). For a cubic B-spline, this is a quintic in parameter s, which we solve by the Newton-Raphson method. Given a solution, s, we then determine the local coordinates of the robot position as:

[1] Not to be confused with the Laplace trasform variable used earlier

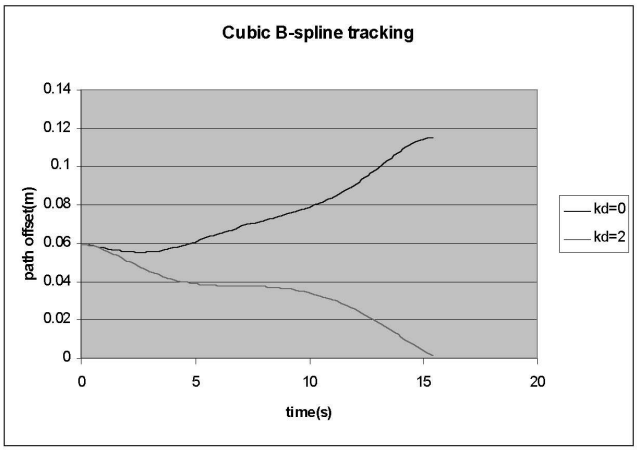

Figure 6: Path offset whilst tracking a cubic B-spline

$$\theta = \theta_g - \tan^{-1} \frac{y'(s)}{x'(s)} \tag{28}$$

$$y = \mathbf{m}.(\hat{\mathbf{t}} X(\mathbf{p_g} - \mathbf{p})) \tag{29}$$

where

$$\hat{\mathbf{t}} = \frac{\mathbf{p}'}{|\mathbf{p}'|} \quad \mathbf{m} = [001] \tag{30}$$

This gives

$$y = \frac{x'(y_g - y) - y'(x_g - x)}{(x'^2 + y'^2)^{\frac{1}{2}}} \tag{31}$$

In addition to the local position of the path, we require the Euclidean path curvature as input to the control system. It can be shown that this is given by

$$\kappa = \frac{y''x' - x''y'}{(x'^2 + y'^2)^{\frac{3}{2}}} \tag{32}$$

To test the control system in following a cubic B-spline, a demand path was set with spline control points of (1,1), (2,1), (3,6) and (8,1). The robot was placed 6cm from the demand path, and parallel to that path, and asked to move at 0.2m/s with k_y, k_θ, k_l set at 1, 4 and 0.1 respectively. The graph labelled "$k_d = 0$" on fig 6 shows the path error increasing due to the combination of the lag caused by the curvature filter and a continuously varying curvature of the demand path.

If rate of change of curvature can be found (wrt time), a proportion of this, $k_d\dot{\kappa}$, may be added to the demand turning curvature to reduce the tracking error. Now it can be shown that

$$\kappa' = \frac{x'(y''' - 3v_s\kappa x'') - y'(x''' + 3v_s\kappa y'')}{v_s^3} \tag{33}$$

where

$$v_s = |\mathbf{p}'| = (x'^2 + y'^2)^{\frac{1}{2}} \tag{34}$$

Curvature rate with respect to time is given as

$$\dot{\kappa} = \frac{\kappa' V_x}{x'} = \frac{\kappa' V_y}{y'} \tag{35}$$

(the term with the largest denominator is evaluated). Demand turning curvature is now given as

$$\kappa_d = -k_y k_\theta y - k_\theta \theta + \kappa_{path} + k_d \dot{\kappa}_{path} \tag{36}$$

The improvement gained in adding the derivative of the path input is shown in figure 6. Future work will determine explicit analytic relations between all four system parameters k_θ, k_y, k_l and k_d for optimum performance over a variety of path types.

8. CONCLUSION

In this paper, the formulation, analysis, and tuning of a mobile robot path tracking system has been described, and it has been tested on paths of varying complexity. Although this control rule is very simple, it has proved successful, because it establishes an intuitive link between the vehicle's degrees of freedom y and θ' relative to its demand path. The velocity independence of key system parameters generates sensible behaviour whereby the vehicle turns more steeply towards its demand path when it is travelling more slowly. In addition, the filtering of curvature demand spreads curvature changes more evenly over the whole process of step disturbance correction.

References

[1] E. Freund and R. Mayr. Nonlinear path control in automated vehicle guidance. *IEEE Trans. on Robotics and Automation*, 13(1):49–60, 1997.

[2] J. Guldner and Utkin V. I. Sliding mode control for gradient tracking and robot navigation using artificial potential fields. *IEEE Trans. on Robotics and Automation*, 11(2):247–254, 1995.

[3] Hongo T. Arakawa H. Sugimoto G. Tange K. and Yamamoto Y. An automatic guidance system od a self controlled vehicle. *IEEE Trans. on Industrial Electronics*, 34(1):5–10, 1987.

[4] O. H. Kim. Optimal steering control of an auto-guided vehicle with two motored wheels. *IEE Trans. of the Institute of Measurement and Control*, 9(2):58–63, 1987.

[5] J. Ma, Y. Kosecka and Sastry S. Vision guided navigation for a nonholonomic mobile robot. *IEEE Trans. on Robotics and Automation*, 15(3):521–536, 1999.

[6] O. J. Sordalen and Canudas de Wit. Exponential control law for a mobile robot: Extension to path following. In *Proc. 1992 IEEE Int. Conf. on Robotics and Automation.*, pages 2158–2163, 1992.

[7] J. M. Yang and Kim J. H. Sliding mode control for trjactory tracking of nonholonomic wheeled vehicles. *IEEE Trans. on Robotics and Automation*, 15(3):578–587, 1999.